.Yazılım Sınama (Test)

- 5.1.Yazılım Sınama
- Sınama (testing); bir programdaki hataları bulmak amacı ile yapılan işlemlerdir.
- Sınama, yazılımın

  a) fonksiyonel,

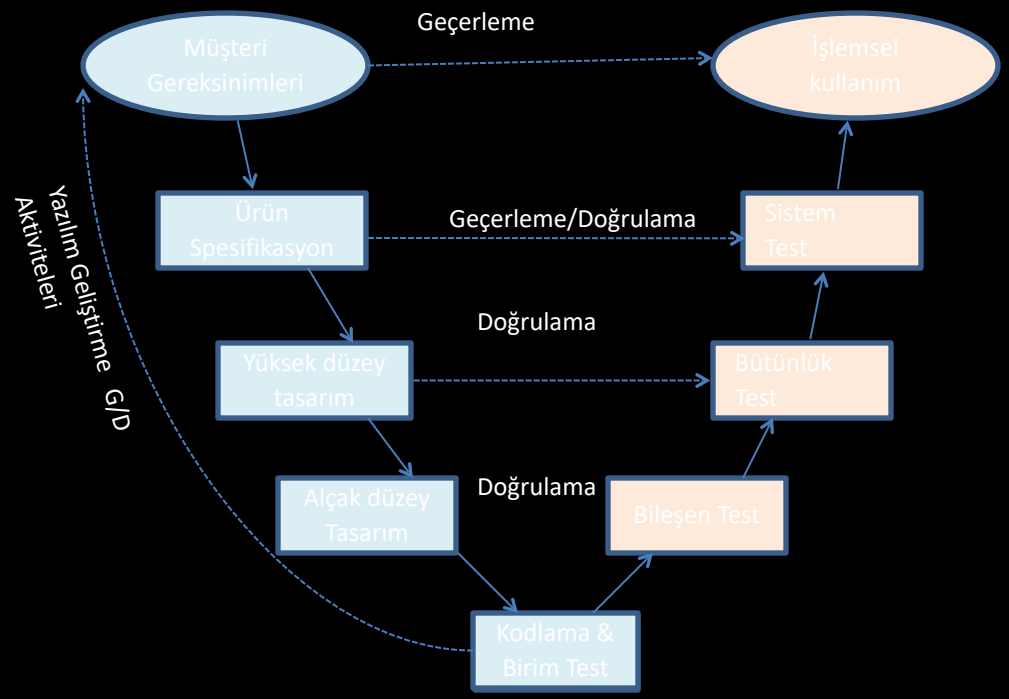
  b) performans,

  c) dayanıklık,

  d) yapısal

bakımlardan yeterliğini denetlemektedir.

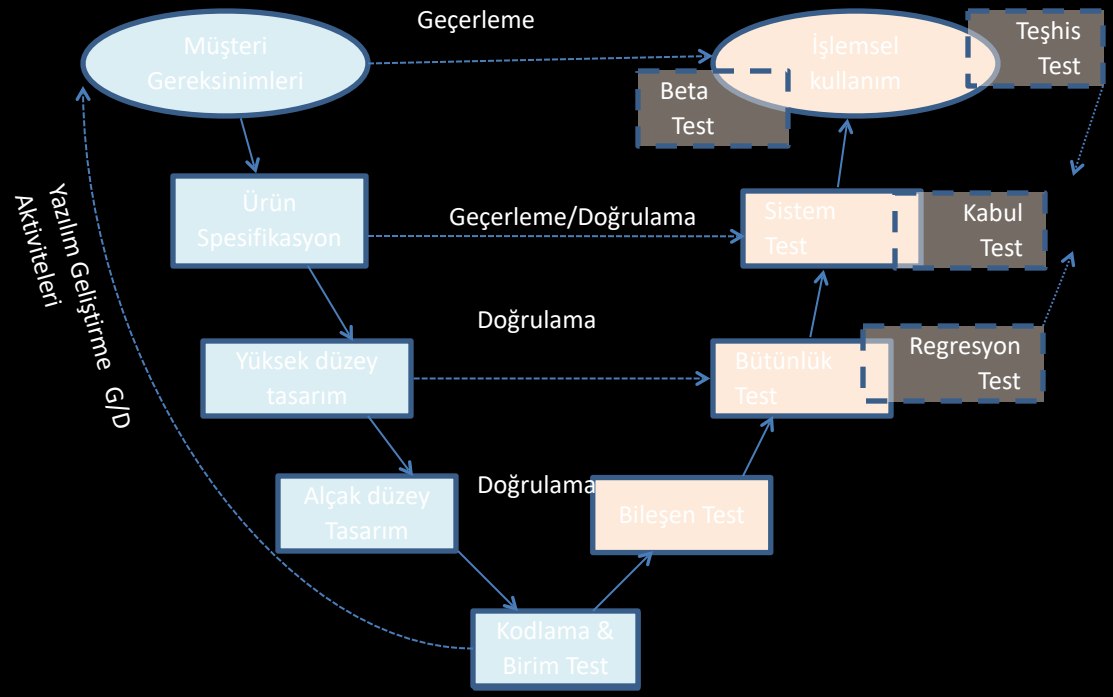
V Model yaklaşımı



Şekil 3.2 Yazılım geliştirme ve Test sürecindeki Geçerleme & Doğrulama Aktiviteleri (V Model yaklaşımı)

5.GEÇERLEME ve DOĞRULAMA TEKNİKLERİ

- 5.0.Bolüm Hedefi
- Dinamik Geçerleme (verification), yazılım test sürecini tanımlama
- Birim test ve Bütünlük test işlemlerini özetlenmesi
- Regresyon testini tanıma
- Saydam kutu Kara kutu Test tiplerini inceleme
- Performans, Dayanıklılık ve Güvenlik Testi olarak Sistem Testini tanımlama

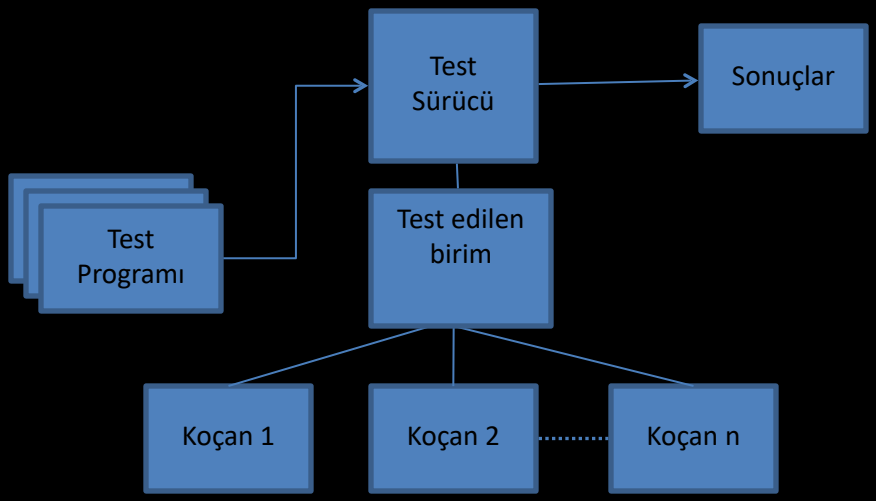


Şekil 5.6 Test sürecindeki Alt Test Adımları (V Model yaklaşımına göre)

5.2.1.Birim Test

- Ünite (birim) testi, yazılım tasarımının en küçük birimi olan modül üzerinde uygulanmaktadır. Ayrıntılı tasarım tanımlarına dayanılarak, modül içerisindeki hataları bulmak üzere, önemli kontrol yolları sınanmaktadır.
- Saydam kutu testi olarak uygulanan bu işlem, çok sayıdaki modül üzerinde, paralel olarak yürütülmektedir.
- Birim testinde; modülün arabirim, lokal veri yapısı, kontrol yapıları arasındaki ana yollar, hata arama yolları ve modül sınırları sınanmaktadır.

Şekil 5.2. Birim Test Ortamı



Birim Test

- Test senaryosu (test case); belirli bir program yolunu işlemek ya da özel bir gereksinime uygunluğu onaylamak amacı ile düzenlenen bir dizi sınama verisinden ve buna ilişkin işlemlerden oluşturulmaktadır.
- Test programlarının geliştirilmesi, diğer yazılımlar gibidir. Geliştirmeye de, test plânı uyarınca ve yazılım tasarımı ile birlikte başlanmalıdır.
- Modülün bağımsız olmaması halinde, sınamada diğer modüller de dikkate alınmalıdır. Bu amaçla her ünite testi için bir “test sürücü”(driver) ve/veya “koçan” (stub) yazılımı geliştirilmektedir.

5.2.2.Bütünleme Testi

- Modüller bağımsız olmayıp, birbirilerine bağlı olmalıdır. Bu bağlantı, “yazılım arabirimi” (software interface) ile sağlanmaktadır.
- Modüllerin birleştirilmesi sırasında veri kaybı, dikkatsizlik nedeni ile birbirini ters etkileme, alt fonksiyonların birleştirilmesiyle beklenilen ana fonksiyonunun gerçekleşmemesi, her birinde göze alınabilen hata toleranslarının eklenerek büyümesi, genel veri yapılarının sorun yaratması söz konusudur. Bu hata ve sorunları bulup gidermek için, modüllerin birleştirilerek ana programın oluşturulmasında, bütünleme testi uygulanmalıdır.

5.2.2.3.Regresyon testi

- **Regresyon Testi** Sınanmış olan bir program veya program parçası üzerinde değişiklik veya ekleme yapılması halinde, tümünün bir kez daha sınanmasıdır.
- Uygulama ortamlarında gerekli değişiklikler ve sabitlemeler yapıldıktan sonra yeniden yapılan testlere regresyon testi denilir.
- Başka bir tanımla, Regresyon Testi, önceden test edilmiş bir yazılımın çeşitli değişiklerden geçtikten sonra da hatasız bir şekilde çalışmasını sağlamak amacıyla yeniden test edilmesi işlemidir.

Sistem Testi

- .**Güvenlik testi:** sistemin zararlı dış müdahalelerden ve bilgi hırsızlığından korunabildiğinin kanıtlanmasıdır.
- **.Dayanıklık (stres) testi;** sistemin miktar, frekans ya da hacım bakımından anormal biçimde yüklenmesi hallerindeki dayanıklığını ölçmek amacı ile düzenlenmektedir.
- **Yetenek (performance) testi**; gerçek zamanlı ve gömülü sistemlerde, yazılım işlem süresinin bilgisayara dayalı sistem ile uyarlığını sınamaktadır. Yeteneğin sınanması, her test basamağında uygulanmaktadır.

7.2.1. Yazılım Bakımı

- Yazılımın bakımı ve onarımı (software maintenance); sonradan görülen hataların düzeltilmesi, yazılımın iyileştirilmesi-uyarlanması ve geliştirilmesi şeklindedir.

Yazılımın bakımı konusundaki işlerin:

- %21'inin hata düzeltme,
- %25'inin iyileştirme,
- %50'sinin uyarlama ve %4'ünün

diğer durumlarda olduğu bildirilmektedir.

7.2.3. Bakım Maliyetlerinin Azaltılması

- Bakım ve onarım giderini en aza indirmek için, yazılım ürününün "bakım ve onarıma elverişli" nitelikte oluşturulması gerekmektedir (maintainability). Bunun için de;
- Yetenekli ve deneyimli yazılım mühendisleri görevlendirmek
- Anlaşılabilir bir sistem yapısı ve kolay işletilebilir bir sistem tasarlamak
- Standart programlama dilleri, işletim sistemleri kullanmak ve belgeleri standart biçimde düzenlemek
- Test programlarından yararlanmak
- Tasarım aşamasında, hata bulma ve düzeltme kolaylıkları sağlamak

gerekmektedir.

Yazılımın bakım ve onarıma elverişliği

Yazılımın bakım ve onarıma elverişliği; yazılımın diğer kalite faktörlerinden olan,

- Sınama kolaylığı
- Basitlik
- Değiştirilebilirlik
- Taşınabilirlik
- Güvenirlik
- Esneklik

özeliklerinin bir bileşkesi olarak ortaya çıkmaktadır.